Compressed decision problems in hyperbolic groups

Derek Holt
University of Warwick, UK
D.F.Holt@warwick.ac.uk

Markus Lohrey
Universität Siegen, Germany
lohrey@eti.uni-siegen.de

Saul Schleimer
University of Warwick, UK
s.schleimer@warwick.ac.uk

Abstract

We prove that the compressed word problem and the compressed simultaneous conjugacy problem are solvable in polynomial time in hyperbolic groups. In such problems, group elements are input as words defined by straight-line programs defined over a finite generating set for the group. We prove also that, for any infinite hyperbolic group G, the compressed knapsack problem in G is NP-complete.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography → Algebraic complexity theory

Keywords and phrases hyperbolic groups, algorithms for compressed words, circuit evaluation problems

Digital Object Identifier 10.4230/LIPIcs.STACS.2019.34

Funding *Markus Lohrey*: The second author has been supported by the DFG research project LO 748/12-1.

Acknowledgements The second author was supported by the DFG research project LO 748/12-1

1 Introduction

Compression techniques in group theory have attracted attention in recent years [9, 10, 30, 36, 37]. Often, algorithms for classical group theoretic problems, such as the word problem or the conjugacy problem, face the problem that huge intermediate words arise during the computation. In some situations, these words are highly compressible; one can then attempt to compute on succinct representatives instead of on the words themselves.

Straight-line programs (SLPs) are a widely-used compression technique for words. An SLP can be seen as a context-free grammar $\mathcal{G}$ that produces a single word denoted $\mathsf{val}(\mathcal{G})$; see Section 4 for a precise definition. The *size* of $\mathcal{G}$ can be defined as the sum of the lengths of the right-hand sides of the productions of $\mathcal{G}$. In fact, the length of $\mathsf{val}(\mathcal{G})$ can be exponential in the size of $\mathcal{G}$, showing that non-trivial compression is possible for SLPs. There are numerous papers in computer science that study the complexity of decision problems for words that are succinctly represented by SLPs; see [29] for a survey. Applications of SLPs in group theory can be traced back to Babai and Szemeredi's reachability theorem for finite groups [2].

In this paper we deal with the so-called compressed word problem for a finitely generated group G, which we define as follows. Suppose that Σ is a finite generating set for G. We assume that Σ is *symmetric*: if a lies in Σ then so does a^{-1}. The *word problem* for G asks whether a given word $w \in \Sigma^*$ represents the group identity of G. This is one of the

fundamental decision problems in group theory as set out by Dehn [8] in 1911. The *compressed word problem* for G is the same problem except that the input word w is represented by an SLP. We also call such inputs *compressed words*.

Clearly, the compressed word problem for a group G is decidable if and only if the word problem for G is decidable. It also seems obvious that the computational complexity of the compressed word problem for G should be more difficult than the word problem itself. This is indeed the case if $\mathsf{P} \neq \mathsf{NP}$; see the discussion on the next page. It is also interesting to note that the compressed word problem for a group G is exactly the *circuit evaluation problem* for G: here the input is a *circuit* (a directed acyclic graph whose nodes are called gates) where input gates are labelled by generators of G and where internal gates compute the product of their inputs. We then ask if a distinguished output gate evaluates to the identity of G. For finite groups, the complexity of the circuit evaluation problem (and hence, the compressed word problem) was clarified in [4]: if G is a finite solvable group, then the compressed word problem for G belongs to the parallel complexity class $\mathsf{DET} \subseteq \mathsf{NC}^2$. Futher, if G is a finite non-solvable group, then the compressed word problem for G is P-complete. This dichotomy naturally motivates the investigation of compressed word problems for general finitely generated groups.

The compressed word problem also has applications for the ordinary (uncompressed) word problem. From techniques very similar to those used in the proof of [40, Theorem 5.2] we may deduce the following: the word problem for a finitely generated subgroup of the automorphism group $\mathsf{Aut}(G)$ is polynomial time reducible to the compressed word problem for G. Similar reductions exist for certain group extensions; see [40, Theorem 4.1] and [30, Theorem 4.8 and 4.9]. This makes groups for which the compressed word problem can be solved in polynomial time interesting. Indeed the class of these groups is quite rich. Let F_n be the free group on n generators. The first result for infinite groups was obtained in [28], where the second author showed that the compressed word problem for F_n is P-complete. This result was used by the third author to show that the word problem for $\mathsf{Aut}(F_n)$ can be solved in polynomial time [40, Theorem 5.2]. This solved an open problem posed by Baumslag [3, Problem (C1)]. Two other important classes of groups in which the compressed word problem can be solved in polynomial time have been found, as follows.

- Virtually special groups; that is, finite extensions of finitely generated subgroups of right-angled Artin groups. Right-angled Artin groups are also known as graph groups or partially commutative groups. Recent work related to three-dimensional topology has shown that the class of virtually special groups is very rich. It contains all Coxeter groups [19], one-relator groups with torsion [42], fully residually free groups [42] (for fully residually free groups, Macdonald [33] independently obtained a polynomial time solution for the compressed word problem), and fundamental groups of hyperbolic 3-manifolds [1].
- Finitely generated nilpotent groups [30]. Here, the compressed word problem even belongs to the parallel complexity class DET [26].

Moreover, for finitely generated linear groups the compressed word problem belongs to the complexity class coRP [30, Theorem 4.15], which implies that there is an efficient randomized polynomial time algorithm that may err with a small probability on negative input instances. On the negative side, it is known that the compressed word problem for every restricted wreath product $G \wr \mathbb{Z}$ with G finitely generated non-abelian is coNP-hard [30, Theorem 4.21]. If G is also finite, then the word problem for $G \wr \mathbb{Z}$ can be easily solved in polynomial time, see also [41]. Assuming $\mathsf{P} \neq \mathsf{NP}$ this gives examples of groups in which the compressed word problem is harder than the word problem. Another interesting result that relates

the compressed word problem to the area of algebraic complexity theory was shown in [30, Theorem 4.16]: The compressed word problem for the linear group $\mathsf{SL}_3(\mathbb{Z})$ is equivalent (up to polynomial time reductions) to polynomial identity testing (that is, the problem whether a circuit over a polynomial ring $\mathbb{Z}[x_1, \ldots, x_n]$ evaluates to the zero polyomial).

In this paper, we prove that the compressed word problem can be solved in polynomial time in every hyperbolic group.[1] Hyperbolic groups have a Cayley graph that satisfies a certain hyperbolicity condition, see Section 3 for a precise definition. Hyperbolic groups are of fundamental importance in geometric group theory. In a certain probabilistic sense, almost all finitely presented groups are hyperbolic [16, 38]. Also from a computational viewpoint, hyperbolic groups have nice properties: it is known that the word problem and the conjugacy problem can be solved in linear time [12, 21]. They also have a nice shortlex automatic structure [11]. We show in Theorem 15 that, from a given SLP $\mathcal{G}$ over the generators of a hyperbolic group G, one can compute in polynomial time an SLP for the shortlex normal form of the word $\mathsf{val}(\mathcal{G})$ (this is the length lexicographically smallest word that represents the same group element as $\mathsf{val}(\mathcal{G})$). Since the shortlex normal form for a word w is the empty word if and only if $w =_G 1$ (here, and in the rest of the paper, we write $u =_G v$ if the words u and v represent the same element of the group G), we obtain the following corollary:

▶ **Corollary 1.** *The compressed word problem for a hyperbolic group can be solved in polynomial time.*

A relatively easy consequence of Corollary 1 is that for every hyperbolic group one can compute in polynomial time the order of the group element that is represented by a given SLP (Corollary 16). In Section 6.2 we consider the *compressed conjugacy problem*: the input consists of SLP-compressed words u, v and it is asked whether there exists a word x with $x^{-1}ux =_G v$. We prove that the compressed conjugacy problem for a hyperbolic group can be solved in polynomial time. For this, we show that the algorithm from [12], which solves the conjugacy problem for a hyperbolic group in linear time, can be implemented in polynomial time for SLP-compressed input words. Based on this algorithm, we then generalise our result on compressed conjugacy to the *compressed simultaneous conjugacy problem*, where the input consists of two finite lists $u_1, \ldots, u_n$ and $v_1, \ldots, v_n$ of SLP-compressed words over the generators of the group G, and it is asked whether there exists a word x with $x^{-1}u_ix =_G v_i$ for $1 \leq i \leq n$. This problem was shown to be solvable in polynomial time for finitely generated nilpotent groups in [34]. In the uncompressed setting, the simultaneous conjugacy problem was shown to be solvable in linear time for hyperbolic groups in [5]. Again, we show that the algorithm in [5] can be implemented in polynomial time for SLP-compressed input words, which yields the following.

▶ **Theorem 2.** *Let G be a hyperbolic group. Then the compressed simultaneous conjugacy problem for G can be solved in polynomial time. Moreover, if the two input lists are conjugate, then we can compute an SLP for a conjugating element in polynomial time.*

The (ordinary) simultaneous conjugacy problem has also been studied for classes of groups other than hyperbolic groups; see for example [25] and the references therein. The SLP-compressed version is important because the word problem for finitely generated subgroups of the outer automorphism group $\mathsf{Out}(G)$ of G can be reduced to the compressed simultaneous conjugacy problem for G [20, Proposition 10]. Note that in [7] it is shown that for a

[1] This result was announced in [30, Theorem 4.12] without proof.

hyperbolic group G, $\mathsf{Aut}(G)$ and hence $\mathsf{Out}(G)$ are finitely generated. Hence, we get the following corollary from our main results.

▶ **Corollary 3.** *For every hyperbolic group G, the word problems for $\mathsf{Aut}(G)$ and $\mathsf{Out}(G)$ can be solved in polynomial time.*

As a byproduct of our algorithm for the compressed simultaneous conjugacy problem we also show that for every hyperbolic group one can compute in polynomial time from a given finite set S of SLP-compressed group elements a finite generating set for the centraliser of S, where every element of this generating set is represented by an SLP. We call this computation problem the *compressed centraliser problem*.

▶ **Theorem 4.** *Let G be a hyperbolic group. Then the compressed centraliser problem for G can be solved in polynomial time.*

Finally, we consider the compressed knapsack problem for a hyperbolic group. In the (ordinary) knapsack problem for a finitely generated group G the input is a list of words $u_1, \ldots, u_n, v$ over the generators of G, and it is asked whether there exist natural numbers $n_1, \ldots, n_k$ such that $v =_G u_1^{n_1} \cdots u_k^{n_k}$. This problem has been studied in [13, 14, 27, 32, 35] for various classes of groups. In [35] it was shown that the knapsack problem for a hyperbolic group can be solved in polynomial time. Recently, this complexity bound was improved to LogCFL (the closure of the class of context-free languages under logspace-reductions) [31]. Moreover, for every non-elementary hyperbolic group (meaning that the group contains a free non-abelian subgroup), knapsack is LogCFL-complete, whereas for elementary hyperbolic groups knapsack belongs to NL (nondeterministic logspace) [31]. In the compressed knapsack problem, the words $u_1, \ldots, u_n, v$ are represented by SLPs. For the special case $G = \mathbb{Z}$ this problem is a variant of the classical knapsack problem for binary encoded integers, which is known to be NP-complete. This makes it interesting to look for groups where the compressed knapsack problem belongs to NP. In [32] it was shown that compressed knapsack for every virtually special group belongs to NP. Here, we prove:

▶ **Theorem 5.** *If G is an infinite hyperbolic group then the compressed knapsack problem for G is NP-complete.*

Full proofs can be found in the arXiv version [22].

2 General notations

Zero is included in the set of natural numbers: that is, $\mathbb{N} = \{0, 1, 2, \ldots\}$. Let Σ be a finite alphabet of symbols. The set of all finite words over Σ is denoted with Σ^*. We use $\varepsilon \in \Sigma^*$ to denote the empty word. Suppose that $w = a_0 a_1 \cdots a_{n-1} \in \Sigma^*$ with $a_i \in \Sigma$. The *length* of w is $|w| = n$. For $0 \leq i \leq n-1$ we define $w[i] = a_i$. For $0 \leq i \leq j \leq n$ we define $w[i:j] = a_i \cdots a_{j-1}$. We use $w[:j]$ to mean $w[0:j]$, the prefix of length j, and we also use $w[i:]$ to mean $w[i:n]$, the suffix of length $n-i$. Note that $w[i:i] = \varepsilon$ and $w = w[:i]w[i:]$ for all $0 \leq i \leq n$. We say that $u \in \Sigma^*$ is a *factor* of $w \in \Sigma^*$ if there exist $x, y \in \Sigma^*$ with $w = xuy$.

Fix a strict linear order $<$ on the alphabet Σ. We extend $<$ to the length-lexicographic order $<_{\mathrm{llex}}$ on Σ^*: for words $u, v \in \Sigma^*$ we have $u <_{\mathrm{llex}} v$ if and only if (i) $|u| < |v|$ or (ii) $|u| = |v|$ and there exist words $x, y, z \in \Sigma^*$ and symbols $a, b \in \Sigma$ such that $a < b$, $u = xay$, and $v = xbz$. Note that $<_{\mathrm{llex}}$ is a well-order on Σ^*. Hence, every non-empty subset $L \subseteq \Sigma^*$ contains a unique smallest element with respect to $<_{\mathrm{llex}}$.

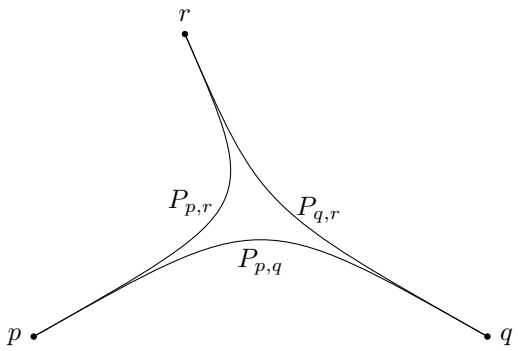

Figure 1 The shape of a geodesic triangle in a hyperbolic group

3 Hyperbolic groups

Let G be a finitely generated group equipped with a finite, symmetric, generating set Σ. The Cayley graph of G with respect to Σ is the directed edge-labelled graph $\Gamma = \Gamma(G)$ with node set G and all edges of the form (g, ga) for $g \in G$ and $a \in \Sigma$. The edge (g, ga) is labelled with the generator a. Note that for every a-labelled edge (g, h), the reversed edge (h, g) is labelled with a^{-1}. We view Γ as a geodesic metric space (the precise definition of a geodesic metric space is not needed in this paper), where every edge (g, ga) is identified with a unit-length interval. The distance between two nodes $p, q \in \Gamma$ is denoted by $d_\Gamma(p, q)$. For $g \in G$ let $|g| := d_\Gamma(1, g)$; so $|g|$ is the length of a shortest word in Σ^* that represents g. For $r \geq 0$, let $\mathcal{B}_r(1) = \{g \in G \colon d_\Gamma(1, g) \leq r\}$.

Given a word $w \in \Sigma^*$, one obtains a unique path $\mathcal{P}[w]$ that starts at 1 and is labelled by the word w. This path ends in the group element represented by w. More generally, for $g \in G$ we denote by $g \cdot \mathcal{P}[w]$ the path that starts at g and is labelled by w. We will mostly consider paths of the form $g \cdot \mathcal{P}[w]$. One views $P := g \cdot \mathcal{P}[w]$ as a continuous mapping $P : [0, n] \to \Gamma$ from the real interval $[0, n]$ to Γ, where $n = |w|$. We say that a path $P : [0, n] \to \Gamma$ is path from $P(0)$ to $P(n)$. A path $P : [0, n] \to \Gamma$ is *geodesic* if $d_\Gamma(P(0), P(n)) = n$. A word $w \in \Sigma^*$ is *geodesic* if the path $\mathcal{P}[w]$ is geodesic, which means that there is no shorter word representing the same group element from G. A word $w \in \Sigma^*$ is *shortlex reduced* if it is the length-lexicographically least word that represents the same group element as w. For this, we have to fix an arbitrary linear order on Σ. Note that if $u = xy$ is shortlex reduced then x and y are shortlex reduced too. For a word $u \in \Sigma^*$ we denote by $\mathsf{shlex}(u)$ the unique shortlex reduced word that represents the same group element as u. Whenever appropriate, we identify elements of $\mathcal{B}_r(1)$ with geodesic words over Σ of length at most r.

A geodesic triangle consists of three points $p, q, r \in \Gamma$ and geodesic paths $P_{p,q}$, $P_{p,r}$, $P_{q,r}$ (the three sides of the triangle), where $P_{x,y}$ is a path from x to y. We call a geodesic triangle *δ-slim* for $\delta \geq 0$, if every point p on one of the three sides has distance at most δ from a point p' belonging to one of the two sides that are opposite to p. The group G is called δ-hyperbolic if every geodesic triangle is δ-slim. Finally, G is hyperbolic, if it is δ-hyperbolic for some $\delta \geq 0$. Figure 1 shows the shape of a geodesic triangle in a hyperbolic group. The property of being hyperbolic is independent of the chosen finite generating set Σ, but the constant δ depends in general on the chosen finite generating set. Finitely generated free groups are for instance 0-hyperbolic, if the generating set is a free basis. The word problem for every hyperbolic group can be decided in real time [21]. Moreover, one can compute $\mathsf{shlex}(w)$ from a given word w in linear time; see [12] where the result is attributed to Shapiro.

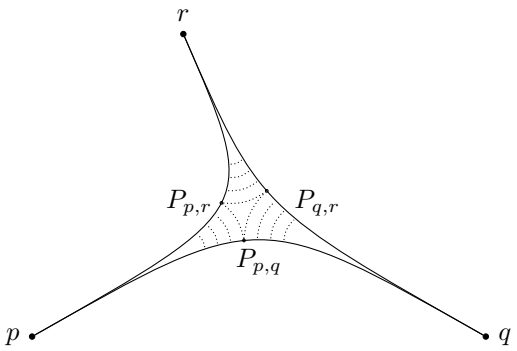

Figure 2 A δ-thin triangle in a hyperbolic group. Dotted lines represent geodesic paths of length at most δ.

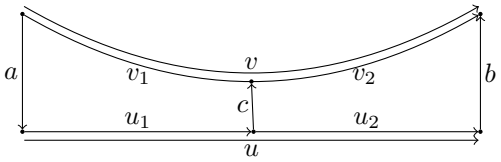

Figure 3 Splitting a geodesic rectangle according to Lemma 7.

We will need an equivalent definition of hyperbolicity in terms of so-called thin triangles. Again, consider three points $p, q, r \in \Gamma$ and let $P_{x,y}$ for $x, y \in \{p, q, r\}$ be a geodesic path from x to y, where $P_{y,x}$ is the path $P_{x,y}$ traversed in the reversed direction. Moreover, let $d_{x,y} = d_\Gamma(x, y)$ be the length of $P_{x,y}$. The three lengths $d_{p,q}$, $d_{p,r}$ and $d_{q,r}$ fulfil the triangle inequality. From this one can deduce real numbers $s_p, s_q, s_r \geq 0$ such that $s_x + s_y = d_{x,y}$ for all $x, y \in \{p, q, r\}$ with $x \neq y$. The geodesic triangle determined by the three sides $P_{p,q}$, $P_{p,r}$, $P_{q,r}$ is called *δ-thin* for $\delta \geq 0$, if for all x, y, z with $x \in \{p, q, r\}$ and $\{y, z\} = \{p, q, r\} \setminus \{x\}$ we have $d_\Gamma(P_{x,y}(t), P_{x,z}(t)) \leq \delta$ for all $t \in [0, s_x]$; see Figure 2. It is well known (see for example [23, Theorem 6.1.3]) that in a δ-hyperbolic group every geodesic triangle is δ'-thin for some constant $\delta' \geq \delta$.

Let us fix a δ-hyperbolic group G with the finite symmetric generating set Σ for the rest of the section, and let Γ be the corresponding geodesic metric space. By choosing δ large enough, we can assume that all geodesic triangles in Γ are δ-thin. We need a few well-known results about hyperbolic groups.

▶ **Lemma 6** (c.f. [11, Theorem 3.4.5]). *The set* $\{\mathsf{shlex}(u) \colon u \in \Sigma^*\}$ *is a regular language.*

The proofs of the following two simple lemmas can be found in [22].

▶ **Lemma 7.** *Let $a, b, u, v \in \Sigma^*$ be geodesic words such that $v =_G aub$ and consider a factorisation $v = v_1 v_2$ with $|v_1| \geq |a| + 2\delta$ and $|v_2| \geq |b| + 2\delta$. Then there exists a factorisation $u = u_1 u_2$ and a geodesic word c with $|c| \leq 2\delta$ such that $v_1 =_G au_1 c$ and $v_2 =_G c^{-1} u_2 b$.*

The situation in Lemma 7 is shown in Figure 3.

▶ **Lemma 8.** *For $i \in \{0, 1, 2\}$ let $P_i : [0, n_i] \to \Gamma$ be geodesic paths such that $P_0(0) = P_1(0)$, $P_0(n_0) = P_2(0)$ and $P_1(n_1) = P_2(n_2)$ (so P_0, P_1, P_2 form a geodesic triangle). Let $j \leq \min\{n_0, n_1\}$ be any integer such that $d_\Gamma(P_0(j), P_1(j)) > \delta$. Then there exist integers $i_0, i_1 \in [0, n_2]$ with $i_0 \leq i_1$, $d_\Gamma(P_0(j), P_2(i_0)) \leq \delta$, and $d_\Gamma(P_1(j), P_2(i_1)) \leq \delta$.*

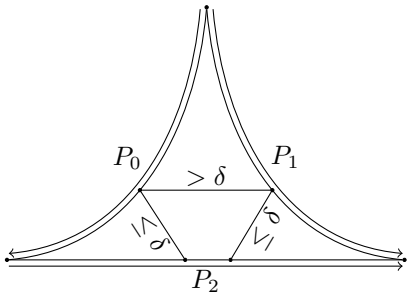

Figure 4 The situation from Lemma 8.

Lemma 8 follows easily from the fact that the geodesic triangle with sides P_0, P_0, and P_1 is thin. The situation is shown in Figure 4.

4 Compressed words and the compressed word problem

A *straight-line program* (SLP for short) over the alphabet Σ is a triple $\mathcal{G} = (V, \rho, S)$, where V is a finite set of variables such that $V \cap \Sigma = \emptyset$, $S \in V$ is the start variable, and $\rho : V \to (V \cup \Sigma)^*$ is a mapping such that the relation $\{(B, A) \in V \times V : B \text{ occurs in } \rho(A)\}$ is acyclic. For the reader familiar with context-free grammars, it might be helpful to view the SLP $\mathcal{G} = (V, \rho, S)$ as the context-free grammar (V, Σ, P, S), where P contains all productions $A \to \rho(A)$ for $A \in V$. The definition of an SLP implies that this context-free grammar derives exactly one terminal word, which will be denoted by $\mathsf{val}(\mathcal{G})$. One can define this string inductively as follows. First, for every $A \in V$ we define $\mathsf{val}_{\mathcal{G}}(A)$. Assume that $\rho(A) = w_0 A_1 w_1 \cdots A_k w_k$ with $k \geq 0$, $w_i \in \Sigma^*$ and $A_i \in V$. Then we define $\mathsf{val}_{\mathcal{G}}(A) = w_0 \mathsf{val}_{\mathcal{G}}(A_1) w_1 \cdots \mathsf{val}_{\mathcal{G}}(A_k) w_k$. Finally, we define $\mathsf{val}(\mathcal{G}) = \mathsf{val}_{\mathcal{G}}(S)$.

The word $\rho(A)$ is also called the *right-hand side* of A. We define the size of the SLP $\mathcal{G} = (V, \rho, S)$ as the total length of all right-hand sides: $|\mathcal{G}| = \sum_{A \in V} |\rho(A)|$. SLPs offer a succinct representation of words that contain many repeated substrings. For instance, the word $(ab)^{2^n}$ can be produced by the SLP $\mathcal{G} = (\{A_0, \ldots, A_n\}, \rho, A_n)$ with $\rho(A_0) = ab$ and $\rho(A_{i+1}) = A_i A_i$ for $0 \leq i \leq n-1$.

Quite often, it is convenient to assume that all right-hand sides are of the form $a \in \Sigma$ or BC with $B, C \in V$. This corresponds to the well-known Chomsky normal form for context-free grammars. There is a simple linear time algorithm that transforms an SLP $\mathcal{G}$ with $\mathsf{val}(\mathcal{G}) \neq \varepsilon$ into an SLP $\mathcal{G}'$ in Chomsky normal form with $\mathsf{val}(\mathcal{G}) = \mathsf{val}(\mathcal{G}')$, see for example [30, Proposition 3.8]. We use the fact that the following algorithmic tasks for SLPs can be solved in polynomial time; see also [30, Proposition 3.9].

- Given an SLP $\mathcal{G}$, compute the length $|\mathsf{val}(\mathcal{G})|$.
- Given an SLP $\mathcal{G}$ and an integer $0 \leq i < |\mathsf{val}(\mathcal{G})|$, compute the symbol $\mathsf{val}(\mathcal{G})[i]$.
- Given an SLP $\mathcal{G}$ and integers $0 \leq i \leq j \leq |\mathsf{val}(\mathcal{G})|$, compute an SLP for $\mathsf{val}(\mathcal{G})[i : j]$.

Also the following two propositions are well-known:

▶ **Proposition 9** (c.f. [6, Lemma 2]). *For a given SLP $\mathcal{G}$ and $n \in \mathbb{N}$, we can compute an SLP $\mathcal{G}_n$ with $\mathsf{val}(\mathcal{G}_n) = \mathsf{val}(\mathcal{G})^n$ in time $O(|\mathcal{G}| + \log n)$.*

▶ **Proposition 10** (c.f. [30, Theorem 3.11]). *Given a deterministic finite state automaton M over the alphabet Σ and an SLP $\mathcal{G}$ over the alphabet Σ, we can determine in polynomial time whether $\mathsf{val}(\mathcal{G})$ is in the language $L(M)$ of M.*

Finally, we need the following fundamental result:

▶ **Theorem 11** (c.f. [39]). *Given two SLPs $\mathcal{G}$ and $\mathcal{H}$, one can check in polynomial time whether* $\mathsf{val}(\mathcal{G}) = \mathsf{val}(\mathcal{H})$.

The *compressed word problem* for a finitely generated group G with the finite symmetric generating set Σ is the following decision problem:
Input: an SLP $\mathcal{G}$ over the alphabet Σ.
Question: does $\mathsf{val}(\mathcal{G})$ represent the group identity of G?
It is an easy observation that the computational complexity of the compressed word problem for G does not depend on the chosen generating set Σ in the sense that if Σ' is another finite symmetric generating set for G, then the compressed word problem for G with respect to Σ is logspace reducible to the compressed word problem for G with respect to Σ' [30, Lemma 4.2]. Therefore we do not have to specify the generating set.

5 The compressed word problem for hyperbolic groups

Fix a δ-hyperbolic group G with the finite symmetric generating set Σ, where $\delta > 0$ is chosen in such a way that all geodesic triangles are δ-thin. We can moreover assume that δ is an integer (later, we want to cut off from a word its prefix and suffix of length a certain multiple of δ). Let us set $\zeta := 2\delta \geq 1$ for the following.

We need an extension of SLPs by two operators: so-called tether operators and cut operators. A TCSLP (T stands for "tethered", C stands for "cut") over the alphabet Σ is a tuple $\mathcal{G} = (V, \rho, S)$, where V is a finite set of variables such that $V \cap \Sigma = \emptyset$, $S \in V$ is the start variable, and ρ is a mapping with domain V such that for every $A \in V$, $\rho(A)$ (the right-hand side of A) is of one of the following forms:
(1) a word $w \in (V \cup \Sigma)^*$
(2) an expression $B[: i]$ or $B[i :]$ with $B \in V$ and $i \in \mathbb{N}$ ($[: i]$ and $[i :]$ are called cut operators),
(3) an expression $B\langle a, b\rangle$ with $B \in V$ and $a, b \in \mathcal{B}_\zeta(1)$ ($\langle a, b\rangle$ is called a tether operator).
Moreover, we require that the relation $\{(B, A) \in V \times V : B \text{ occurs in } \rho(A)\}$ is acyclic. The reflexive and transitive closure of this relation is denoted with $\leq_{\mathcal{G}}$. We evaluate variables of type (1) as for SLPs. If $\rho(A) = B[: i]$ or $\rho(A) = B[i :]$ then we define $\mathsf{val}_{\mathcal{G}}(A) = \mathsf{val}_{\mathcal{G}}(B)[: i]$ or $\mathsf{val}_{\mathcal{G}}(A) = \mathsf{val}_{\mathcal{G}}(B)[i :]$, respectively. Finally, if $\rho(A) = B\langle a, b\rangle$ we define $\mathsf{val}_{\mathcal{G}}(A) := \mathsf{shlex}(a\, \mathsf{val}_{\mathcal{G}}(B)\, b)$. The reader might ask what happens if $i > |\mathsf{val}_{\mathcal{G}}(B)|$ in case $\rho(A) = B[: i]$ or $\rho(A) = B[i :]$. This will not occur in the TCSLPs constructed in this paper.

For convenience we will also allow more complex right-hand sides $\rho(A)$ such as for instance $(B[: i]\langle a, b\rangle)(C[j :]\langle c, d\rangle)$. We define the *size* of such a right-hand side as the total number of occurrences of symbols from $\Sigma \cup V$ in the right-hand side. The size of $\mathcal{G}$ is obtained by taking the sum over all variables. Note that the numbers $i \in \mathbb{N}$ in cut operators do not contribute to the size of a TCSLP. This does not cause any problems. If one encodes these numbers in binary notation and adds the lengths of these encodings to the size of an TCSLP, then this would only increase the size by a constant factor.

If right-hand sides of type (2) do not occur, we speak of a TSLP and if right-hand sides of type (3) do not occur, we speak of a CSLP. CSLPs are also known as *composition systems* and have been studied before, see for example [18]. In [28], CSLPs are used in the polynomial time algorithm for the compressed word problem of a free group.

We say that the TCSLP $\mathcal{G}$ is *shortlex*, if for every variable A, the word $\mathsf{val}_{\mathcal{G}}(A)$ is shortlex reduced. Note that right-hand sides of type (1) may lead to words that are not

shortlex reduced, since the concatenation of two shortlex reduced words is not necessarily shortlex reduced. The goal of this section is to compute from a given shortlex TCSLP $\mathcal{G}$ in polynomial time an SLP $\mathcal{G}'$ such that $\mathsf{val}(\mathcal{G}) =_G \mathsf{val}(\mathcal{G}')$. In a first step we will present such a transformation only for TSLPs. In a second step, we will transform a shortlex TCSLP into an equivalent TSLP. This second step is inspired by the polynomial time transformation of a CSLP into an equivalent SLP from [18].

For a variable A, we define the height, $\mathsf{height}(A)$ for short, inductively by

$$\mathsf{height}(A) = \max\{\mathsf{height}(B) + 1 \colon B \in V \text{ occurs in } \rho(A)\},$$

where $\max \emptyset = 0$. Let $\mathsf{height}(\mathcal{G}) = \mathsf{height}(S)$; it is the length of a longest chain in the partial order $\leq_{\mathcal{G}}$ that ends in S.

▶ **Lemma 12.** *Given a TSLP $\mathcal{G}$ over the alphabet Σ, we can check in polynomial time whether $\mathcal{G}$ is shortlex. Moreover, if $\mathcal{G}$ is shortlex, then we can compute in polynomial time an SLP $\mathcal{G}'$ over G such that $\mathsf{val}(\mathcal{G}) =_G \mathsf{val}(\mathcal{G}')$.*

Proof. Let $\mathcal{G} = (V, \rho, S)$. In the same way as for SLPs, we can assume that all right-hand sides from $(V \cup \Sigma)^*$ are of the form $a \in \Sigma$ or BC with $B, C \in V$ (variables with right-hand side ε can be eliminated). Let $\mu = \mathsf{height}(\mathcal{G})$. We will transform $\mathcal{G}$ into the desired SLP $\mathcal{G}'$. This will be done by a bottom-up process; that is we consider the variables in $\mathcal{G}$ in order of increasing height. If $\mathcal{G}$ is not shortlex, we will detect this during the transformation.

For a variable A, we define the tether-height, $\mathsf{theight}(A)$ for short, inductively as follows:

- if $\rho(A) = a$, then $\mathsf{theight}(A) = 0$,
- if $\rho(A) = BC$, then $\mathsf{theight}(A) = \max\{\mathsf{theight}(B), \mathsf{theight}(C)\}$, and
- if $\rho(A) = B\langle s, t\rangle$ then $\mathsf{theight}(A) = \mathsf{theight}(B) + 1$.

By removing unused variables, we can assume that S has maximal height and maximal tether height among all variables. For a nonterminal A we define $\eta_A := \mathsf{theight}(S) - \mathsf{theight}(A) + 1 > 0$.

Consider a nonterminal A. Since we are processing the variables in order of increasing height, we can assume that for all $B <_{\mathcal{G}} A$ the word $\mathsf{val}_{\mathcal{G}}(B)$ is shortlex reduced. Let $w := \mathsf{val}_{\mathcal{G}}(A)$. If $|w| \leq 16\zeta\eta_A + 2\zeta$ then we will explicitly compute the word w in the process of defining the SLP $\mathcal{G}'$. Otherwise, we will compute explicitly words ℓ_A, r_A such that $w = \ell_A w' r_A$ for some word w' of length at least 2ζ. The words ℓ_A and r_A will satisfy the length constraints $8\zeta\eta_A \leq |\ell_A|, |r_A| \leq 8\zeta\eta_A + 2\zeta\mathsf{height}(A)$. Moreover, in the latter case, the SLP $\mathcal{G}'$ will contain variables $A'_{a,b}$ for all $a, b \in \mathcal{B}_\zeta(1)$ such that $\mathsf{val}_{\mathcal{G}'}(A'_{a,b}) = \mathsf{shlex}(aw'b)$. The $A'_{a,b}$, together with a start variable S', which will be added at the end of the process, will be the only variables that we include in the SLP $\mathcal{G}'$. All of the words that we compute and store, such as the ℓ_A and r_A, are to enable us to carry out the necessary computations, and are not stored as part of $\mathcal{G}'$.

We now make a case distinction on the form of the right-hand side $\rho(A)$. We only consider the most difficult case that $\rho(A) = B\langle a, b\rangle$ for $a, b \in \mathcal{B}_\zeta(1)$.

Let $u := \mathsf{val}_{\mathcal{G}}(B)$ and $v := \mathsf{val}_{\mathcal{G}}(A) = \mathsf{shlex}(aub)$. The word u is shortlex reduced by assumption, and v is shortlex reduced by definition. Let $\eta = \eta_B$. We have $\eta_A = \eta - 1 \geq 1$. If $|u| \leq 16\zeta\eta + 2\zeta$ then we have explicitly computed the word u. We explicitly compute the word $v = \mathsf{shlex}(aub)$, and then distinguish the cases $|v| \leq 16\zeta\eta + 2\zeta$ and $|v| > 16\zeta\eta + 2\zeta$. In the first case, there is nothing to do. If $|v| > 16\zeta\eta + 2\zeta$, we factorise v as $v = \ell_A v' r_A$ with $|\ell_A| = |r_A| = 8\zeta\eta$, and thus $|v'| \geq 2\zeta$. We can compute for all $a, b \in \mathcal{B}_\zeta(1)$ the word $\mathsf{shlex}(av'b)$ and set $\rho'(A'_{a,b}) = \mathsf{shlex}(av'b)$.

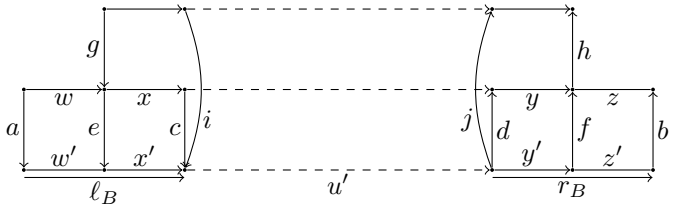

Figure 5 The situation from the proof of Lemma 12. Dotted lines represent words that are given by SLPs.

Now assume that $|u| > 16\zeta\eta + 2\zeta$. We have computed words ℓ_B, r_B such that $8\zeta\eta \leq |\ell_B|, |r_B| \leq 8\zeta\eta + 2\zeta\mathsf{height}(B)$ and $u = \ell_B u' r_B$ for a word u' of length at least 2ζ. Moreover, we have already defined variables $B'_{c,d}$ for all $c, d \in \mathcal{B}_\zeta(1)$, which produce $\mathsf{shlex}(cu'd)$.

Using Proposition 10 we check in polynomial time for all $c, d \in \mathcal{B}_\zeta(1)$ whether the word

$$\mathsf{shlex}(a\ell_B c^{-1})\mathsf{val}_{\mathcal{G}'}(B'_{c,d})\mathsf{shlex}(d^{-1}r_B b) = \mathsf{shlex}(a\ell_B c^{-1})\mathsf{shlex}(cu'd)\mathsf{shlex}(d^{-1}r_B b)$$

is shortlex reduced, in which case it is $\mathsf{shlex}(a\ell_B u' r_B b) = \mathsf{shlex}(aub) = v$; see Figure 5. By Lemma 7, there must exist such $c, d \in \mathcal{B}_\zeta(1)$. Let $s = \mathsf{shlex}(a\ell_B c^{-1})$ and $t = \mathsf{shlex}(d^{-1}r_B b)$. By the triangle inequality, these words have length at least $8\zeta\eta - 2\zeta$. Hence we can factorise these words as $s = wx$ and $t = yz$ with $|w| = |z| = 8\zeta(\eta - 1) = 8\zeta\eta_A \geq 8\zeta$. The words x and y have length at least 6ζ. We set $\ell_A := w$ and $r_A := z$. These words satisfy the required bounds on their lengths. Note that $\mathsf{val}_{\mathcal{G}}(A) = \mathsf{shlex}(aub) = \ell_A x\,\mathsf{shlex}(cu'd) y r_A$ and $|x\,\mathsf{shlex}(cu'd)y| \geq 12\zeta \geq 2\zeta$.

It remains to define the right-hand sides of the variables $A'_{g,h}$ for all $g, h \in \mathcal{B}_\zeta(1)$. Let us fix $g, h \in \mathcal{B}_\zeta(1)$. The lower bounds on the lengths of w, x, y, z allow us to apply Lemma 7 to the geodesic rectangles with sides a, ℓ_B, c, wx and d, r_B, b, yz, respectively (all of these words have been computed explicitly). We can compute in polynomial time $e, f \in \mathcal{B}_\zeta(1)$ and factorisations $\ell_B = w'x'$, $r_B = y'z'$ as shown in Figure 5. By the triangle inequality, the words x' and y' must have length at least 4ζ. Now consider the geodesic rectangle with sides $x'u'y'$, $\mathsf{shlex}(ge)$, $\mathsf{shlex}(fh)$, and $\mathsf{shlex}(gex'u'y'fh)$. Since $|x'|, |y'| \geq 4\zeta$ and $|\mathsf{shlex}(ge)|, |\mathsf{shlex}(fh)| \leq 2\zeta$, we can apply Lemma 7 again: There must exist $i, j \in \mathcal{B}_\zeta(1)$ such that the word

$$\mathsf{shlex}(gex'i^{-1})\,\mathsf{val}_{\mathcal{G}'}(B'_{i,j})\,\mathsf{shlex}(j^{-1}y'fh) = \mathsf{shlex}(gex'i^{-1})\,\mathsf{shlex}(iu'j)\,\mathsf{shlex}(j^{-1}y'fh)$$

is shortlex reduced, in which case the above word is $\mathsf{shlex}(gex'u'y'fh)$. Using Proposition 10, we can compute such $i, j \in \mathcal{B}_\zeta(1)$ in polynomial time. We finally define the right-hand side of $A'_{g,h}$ as $\rho'(A'_{g,h}) = \mathsf{shlex}(gex'i^{-1})\,B'_{i,j}\,\mathsf{shlex}(j^{-1}y'fh)$.

This concludes the definition of the right-hand sides for the variables $A'_{a,b}$. We complete the definition of $\mathcal{G}'$ by adding a start variable S' to $\mathcal{G}'$ and setting $\rho'(S') = \ell_S S'_{1,1} r_S$. ◀

The next lemma generalises Lemma 12 to TCSLPs.

▶ **Lemma 13.** *Given a TCSLP $\mathcal{G}$ over the alphabet Σ, we can check in polynomial time whether $\mathcal{G}$ is shortlex. Moreover, if $\mathcal{G}$ is shortlex, then we can compute in polynomial time an SLP $\mathcal{G}'$ over G such that $\mathsf{val}(\mathcal{G}) =_G \mathsf{val}(\mathcal{G}')$.*

Proof sketch. The idea of the proof is taken from [18], where it is shown that a CSLP can be transformed in polynomial time into an equivalent SLP. Let $\mathcal{G} = (V, \rho, S)$ be the input TCSLP. We can assume that all right-hand sides from $(V \cup \Sigma)^*$ are of the form $a \in \Sigma$ or BC with $B, C \in V$. By Lemma 12 it suffices to transform $\mathcal{G}$ into an equivalent TSLP. Let

$\mu = \mathsf{height}(\mathcal{G})$. Consider a variable A such that $\rho(A) = B[:i]$; the case that $\rho(A) = B[i:]$ can be dealt with analogously. We can assume that $i \leq |\mathsf{val}_{\mathcal{G}}(B)|$ (this will be true for the TCSLP constructed in the proof of Theorem 15 below). By considering the variables in order of increasing height, we can moreover assume that no cut operator occurs in the right-hand side of any variable $C <_{\mathcal{G}} A$. We then push the operator in $\rho(A)$ towards smaller (with respect to $<_{\mathcal{G}}$) variables. Thereby we add at most μ new variables to the TCSLP. Moreover the height of the TCSLP after the cut elimination is still bounded by μ. Hence, the final TSLP has at most $\mu \cdot |V|$ variables. In addition, every right-hand side of the final TSLP will have length at most $2\zeta + 1$, so its size will be polynomially bounded. ◀

▶ **Lemma 14.** *Given shortlex TCSLPs $\mathcal{G}_0$ and $\mathcal{G}_1$ such that $\mathsf{val}(\mathcal{G}_i)$ represents the group element g_i, we can check in polynomial time, whether $d_\Gamma(g_0, g_1) \leq \delta$. Moreover, if this is true, we can compute $a \in \mathcal{B}_\delta(1)$ such that $g_0 a =_G g_1$.*

Proof. For all $a \in \mathcal{B}_\delta(1)$ we compute, by adding one new variable to $\mathcal{G}_0$, a shortlex TCSLP $\mathcal{G}_{0,a}$ for $\mathsf{shlex}(\mathsf{val}(\mathcal{G}_0)a)$. Using Lemma 13 and Theorem 11 we can check in polynomial time whether $\mathsf{val}(\mathcal{G}_{0,a}) = \mathsf{val}(\mathcal{G}_1)$, which is equivalent to $g_0 a =_G g_1$. ◀

Finally, we can prove the main technical result of this section:

▶ **Theorem 15.** *From a given SLP $\mathcal{G}$ over the alphabet Σ we can compute in polynomial time an SLP $\mathcal{G}'$ for $\mathsf{shlex}(\mathsf{val}(\mathcal{G}))$.*

Proof. By Lemma 13 it suffices to compute in polynomial time a shortlex TCSLP $\mathcal{G}'$ for $\mathsf{shlex}(\mathsf{val}(\mathcal{G}))$. For this, we process $\mathcal{G}$ bottom-up; that is, we consider the variables in order of increasing height. Assume that $\mathcal{G} = (V, \rho, S)$ and that $\mathcal{G}$ is in Chomsky normal form. The TCSLP $\mathcal{G}'$ will contain all variables from V plus some auxiliary variables. Let us write $\mathcal{G}' = (V', \rho', S)$. For every variable $A \in V$ we will have $\mathsf{val}_{\mathcal{G}'}(A) = \mathsf{shlex}(\mathsf{val}_{\mathcal{G}}(A))$. Consider a variable $A \in V$ and assume that, for all variables $B <_{\mathcal{G}} A$, we have already defined $\rho'(B)$ in such a way that $\mathsf{val}_{\mathcal{G}'}(B) = \mathsf{shlex}(\mathsf{val}_{\mathcal{G}}(B))$.

If $\rho(A) = a \in \Sigma$ then we set $\rho'(A) := \mathsf{shlex}(a)$. Now assume that $\rho(A) = BC$. Thus we have already defined TCSLPs for the words $u := \mathsf{shlex}(\mathsf{val}_{\mathcal{G}}(B))$ and $v := \mathsf{shlex}(\mathsf{val}_{\mathcal{G}}(C))$. Moreover, by Lemma 13 we can transform these TCSLPs into SLPs. Using these SLPs, we can compute the lengths $m = |u|$ and $n = |v|$. If $m = 0$ or $n = 0$, then we set $\rho'(A) := C$ or $\rho'(A) := B$, respectively. So let us assume that m and n are both non-zero. Moreover, we only consider the case that $m \leq n$; the other case is symmetric. From the SLP for u we can compute an SLP for u^{-1}. Consider the geodesic paths $P_0 := \mathcal{P}[u^{-1}]$ and $P_1 := \mathcal{P}[v]$. Using Lemma 14 we can check whether $d_\Gamma(P_0(m), P_1(m)) \leq \delta$.

Case 1. $d_\Gamma(P_0(m), P_1(m)) \leq \delta$. In this case, we can compute by Lemma 14 a word a of length at most δ such that $a =_G uv[:m]$. The situation is shown in Figure 6 on the left. We set $\rho'(A) := C[m:]\langle a, 1\rangle$.

Case 2. $d_\Gamma(P_0(m), P_1(m)) > \delta$. Using binary search, we compute an integer $i \in [0, m-1]$ such that $d_\Gamma(P_0(i), P_1(i)) \leq \delta$ and $d_\Gamma(P_0(i+1), P_1(i+1)) > \delta$. For this we store an interval $[p, q] \subseteq [0, m]$ such that $p < q$, $d_\Gamma(P_0(p), P_1(p)) \leq \delta$ and $d_\Gamma(P_0(q), P_1(q)) > \delta$. Initially, we set $p = 0$ and $q = m$, and we stop if $q = p + 1$. In each iteration, we compute $r = \lceil (p+q)/2 \rceil$ and check, using Lemma 14, whether $d_\Gamma(P_0(r), P_1(r)) \leq \delta$ or $d_\Gamma(P_0(r), P_1(r)) > \delta$. In the first case we set $p := r$ and do not change q, and in the second case we set $q := r$ and do not change p. Hence, in each iteration the size of the interval $[p, q]$ is roughly halved. Therefore, the binary search stops after $\mathcal{O}(\log(m))$ iterations, which is polynomial in the input length.

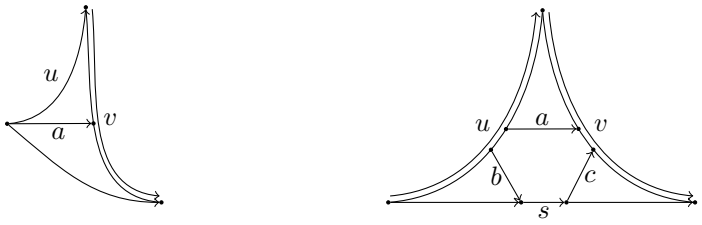

Figure 6 Case 1 (left) and 2 (right) from the proof of Theorem 15.

In addition to the position i, we can also compute $a \in \mathcal{B}_\delta(1)$ that labels a path from $P_0(i)$ to $P_1(i)$.

Let P_2 be the unique geodesic path from $P_0(m)$ to $P_1(n)$ that is labelled with a shortlex reduced word. Note that this path is labelled with $\mathsf{shlex}(uv)$. By Lemma 8 there exist $i_0 \leq i_1$ such that $d_\Gamma(P_0(i+1), P_2(i_0)) \leq \delta$ and $d_\Gamma(P_1(i+1), P_2(i_1)) \leq \delta$. We therefore iterate over all $b, c \in \mathcal{B}_\delta(1)$, compute the word $s := \mathsf{shlex}(b^{-1}u[m-i-1]av[i]c^{-1})$ explicitly, and check whether the word

$$\mathsf{shlex}(u[:m-i-1]b)\, s\, \mathsf{shlex}(cv[i+1:]) \tag{1}$$

is shortlex reduced too, in which case it is $\mathsf{shlex}(uv)$. This can be done using Proposition 10 and using the fact that SLPs for u and v are available. From these SLPs we can compute TCSLPs for $\mathsf{shlex}(u[:m-i-1]b)$ and $\mathsf{shlex}(cv[i+1:])$, which can be transformed into SLPs using Lemma 13. It is guaranteed by Lemma 8 that we will find $b, c \in \mathcal{B}_\delta(1)$ such that the word in (1) is shortlex reduced. For these b, c we finally set $\rho'(A) := (B[:m-i-1]\langle 1, b\rangle)\ s\ (C[i+1:]\langle c, 1\rangle)$. This concludes the proof of the theorem. ◀

A word $w \in \Sigma^*$ represents the group identity if and only if $\mathsf{shlex}(w) = \varepsilon$. Hence, Corollary 1 from the introduction follows directly from Theorem 15.

6 Further compressed decision problems

6.1 Computing the order of a compressed group element

An easy consequence of Corollary 1 is the following result:

▶ **Corollary 16.** *Let G be a hyperbolic group G with the finite symmetric generating set Σ. From a given SLP $\mathcal{G}$ over the alphabet Σ one can compute in polynomial time the order (an element from $\mathbb{N} \cup \{\infty\}$) of the group element represented by $\mathsf{val}(\mathcal{G})$.*

Proof. Let $\mathcal{G}$ be an SLP over the alphabet Σ. It is known that every hyperbolic group has a finite number of conjugacy classes of finite subgroups, and hence that there is a bound on the order of its finite subgroups [23, Theorem 6.8.4]. So there exists a constant $c = c(G)$ such that the order of every element $g \in G$ belongs to $\{1, \ldots, c, \infty\}$. Hence, in order to compute the order of (the group element represented by) $\mathsf{val}(\mathcal{G})$, it suffices to check whether $\mathsf{val}(\mathcal{G})^k =_G 1$ for any $1 \leq k \leq c$. By Corollary 1, this can be done in polynomial time. ◀

6.2 Compressed conjugacy and centralisers

Let G be a finitely generated group G with a fixed finite symmetric generating set for G. For group elements $g, h \in G$ we use the standard abbreviation $g^h = h^{-1}gh$, which is extended to lists $\mathcal{L} = (g_1, \ldots, g_k)$ with $g_i \in G$ by $\mathcal{L}^h = (g_1^h, \ldots, g_k^h)$. We extend these definitions to

words over Σ in the obvious way. The *compressed conjugacy problem* for G is the following problem:

Input: SLPs $\mathcal{G}$ and $\mathcal{H}$ over the alphabet Σ.

Question: Do $\mathcal{G}$ and $\mathcal{H}$ represent conjugate elements in G? That is, does there exist $g \in G$ with $\mathsf{val}(\mathcal{G})^g =_G \mathsf{val}(\mathcal{H})$?

More generally, we can define the *compressed simultaneous conjugacy problem* for G:

Input: Finite lists $\mathcal{L}_\mathcal{G} := (\mathcal{G}_1, \ldots, \mathcal{G}_k)$ and $\mathcal{L}_\mathcal{H} := (\mathcal{H}_1, \ldots, \mathcal{H}_k)$ of SLPs over the alphabet Σ.

Question: Do $\mathcal{L}_\mathcal{G}$ and $\mathcal{L}_\mathcal{H}$ represent conjugate lists of elements in G? That is, does there exist $g \in G$ with $\mathsf{val}(\mathcal{G}_i)^g =_G \mathsf{val}(\mathcal{H}_i)$ for all $1 \leq i \leq k$?

In the case when the answer to either of these questions is positive, we might also want to compute an SLP for an element $g \in G$ that conjugates $\mathsf{val}(\mathcal{G})$ to $\mathsf{val}(\mathcal{H})$ or $\mathcal{L}_\mathcal{G}$ to $\mathcal{L}_\mathcal{H}$.

The *compressed centraliser problem* for G is the following computation problem:

Input: A finite list $(\mathcal{G}_1, \ldots, \mathcal{G}_k)$ of SLPs over G.

Output: A finite list of SLPs $(\mathcal{H}_1, \ldots, \mathcal{H}_l)$ such that $\{\mathsf{val}(\mathcal{H}_1), \ldots, \mathsf{val}(\mathcal{H}_l)\}$ is a generating set for the centraliser of the group elements represented by $\mathsf{val}(\mathcal{G}_1), \ldots, \mathsf{val}(\mathcal{G}_k)$.

The proofs of Theorems 2 and 4 from the introduction can be found in the full version [22]. A linear-time algorithm for solving the conjugacy problem of a hyperbolic group G is described in [12, Section 3]. This was generalised in [5] to a linear-time algorithm for the (uncompressed) simultaneous conjugacy problem and the centraliser problem. We show in [22] that essentially the same algorithms (modulo applications of Theorem 15) can be used to solve the compressed (simultaneous) conjugacy problem for G and the compressed centraliser problem in polynomial time.

6.3 Compressed knapsack

Let G be a finitely generated group with the finite symmetric generating set Σ. A *knapsack expression* over G is a rational expression of the form $E = v_0 u_1^* v_1 u_2^* v_2 \cdots u_k^* v_k$ with $k \geq 0$ and $u_i, v_i \in \Sigma^*$. A solution for E is a tuple $(n_1, n_2, \ldots, n_k) \in \mathbb{N}^k$ of natural numbers such that $v_0 u_1^{n_1} v_1 u_2^{n_2} v_2 \cdots u_k^{n_k} v_k =_G 1$. The *length* of E is defined as $|E| = |v_0| + \sum_{i=1}^k |u_i| + |v_i|$. The knapsack problem for G is the following decision problem:

Input: A knapsack expression E over G.

Question: Does E has a solution?

In [35] it was shown that the knapsack problem for a hyperbolic group can be solved in polynomial time. A crucial step in the proof for this fact is the following result, which is of independent interest:

▶ **Theorem 17** (c.f. [35]). *For every hyperbolic group G there exists a polynomial $p(x)$ such that the following holds: if a knapsack expression $E = v_0 u_1^* v_1 u_2^* v_2 \cdots u_k^* v_k$ over G has a solution then it has a solution $(n_1, \ldots, n_k) \in \mathbb{N}^k$ such that $n_i \leq p(|E|)$ for all $1 \leq i \leq k$.*

Let us now consider the *compressed knapsack problem* for G. It is defined in the same way as the knapsack problem, except that the words $u_i, v_i \in \Sigma^*$ are given by SLPs. The compressed knapsack problem for $\mathbb{Z}$ is NP-complete [17, Proposition 4.1.1]. In fact, this problem corresponds to a variant of the classical knapsack problem for binary encoded integers (for an integer z, it is easy to construct in polynomial time from the binary encoding of z an SLP over the symmetric generating set $\{a, a^{-1}\}$ of $\mathbb{Z}$ which evaluates to a^z or to $(a^{-1})^{-z}$). Using this fact, Corollary 1 and Theorem 17, we can easily deduce Theorem 5 from the introduction, as follows.

Proof of Theorem 5. Consider a knapsack expression $E = v_0 u_1^* v_1 u_2^* v_2 \cdots u_k^* v_k$ over G, where the u_i and v_i are given by SLPs $\mathcal{G}_i$ and $\mathcal{H}_i$, respectively. We then have $|u_i| \leq 3^{|\mathcal{G}_i|/3}$ and $|v_i| \leq 3^{|\mathcal{H}_i|/3}$; see the proof of Lemma 1 in [6] (these bounds on the lengths of the u_i and v_i do not assume that the $\mathcal{G}_i$ and $\mathcal{H}_i$ are in Chomsky normal form). Let $N := |\mathcal{H}_0| + \sum_{i=1}^{k} (|\mathcal{G}_i| + |\mathcal{H}_i|)$ be the input length. By Theorem 17, there exists a polynomial $p(x)$ such that E has a solution if and only if it has a solution $(n_1, \ldots, n_k) \in \mathbb{N}^k$ such that $n_i \leq p(|E|)$ for all $1 \leq i \leq k$. We obtain a bound of the form $2^{\mathcal{O}(N)}$ on the n_i. Hence, we can guess a tuple $(n_1, \ldots, n_k) \in \mathbb{N}^k$ with all n_i bounded by $2^{\mathcal{O}(N)}$ and then check whether it is a solution of E. The latter can be done in polynomial time by constructing from the SLPs $\mathcal{G}_i$ and $\mathcal{H}_i$ an SLP $\mathcal{G}$ for $v_0 u_1^{n_1} v_1 u_2^{n_2} v_2 \cdots u_k^{n_k} v_k$ using Proposition 9. Finally, we check in polynomial time whether $\mathsf{val}(\mathcal{G}) =_G 1$ using Corollary 1.

The second statement of Theorem 5 follows from the well known fact that every infinite hyperbolic group contains a copy of $\mathbb{Z}$ together with the above mentioned result for $\mathbb{Z}$. ◀

7 Conclusion and open problems

We proved that for every hyperbolic group G, several compressed decision problems (where input words are represented by straight-line programs) can be solved in polynomial time, namely the compressed versions of the following problems: the word problem, computing the order of a group element, the simultaneous conjugacy problem, computing the centralizer of a finite set of group elements, and the knapsack problem.

An important open problem is the precise complexity of the compressed word problem for finitely generated linear groups. We mentioned in the introduction that the compressed word problem for every finitely generated linear group belongs to the complexity class coRP. It is open, whether this upper bound can be improved to P for every finitely generated linear group. This is a very difficult question: as mentioned in the introduction, the compressed word problem for the finitely generated linear group $\mathsf{SL}_3(\mathbb{Z})$ is equivalent (up to polynomial time reductions) to polynomial identity testing. The precise complexity of the latter problem is an outstanding open problem in algebraic complexity theory that is tightly related to lower bounds in circuit complexity theory [24]. But there are many interesting subclasses of finitely generated linear groups, for which it is open whether a polynomial time algorithm for the compressed word problem exists. Let us mention braid groups and Baumslag-Solitar groups $\mathsf{BS}(1, p)$ (for $p \geq 2$) in this context.

Another interesting class of groups, where compressed decision problems have not been considered in depth so far are automaton groups. A concrete open problem is the complexity of the compressed word problem for the Grigorchuk group. The (uncompressed) word problem for the Grigorchuk group can be solved in deterministic logarithmic space [15].

References

1 Ian Agol. The virtual Haken conjecture. *Documenta Mathematica*, 18:1045–1087, 2013. With an appendix by Ian Agol, Daniel Groves, and Jason Manning.

2 László Babai and Endre Szemerédi. On the complexity of matrix group problems I. In *Proceedings of the 25th Annual Symposium on Foundations of Computer Science, FOCS 1984*, pages 229–240. IEEE Computer Society, 1984.

3 Gilbert Baumslag, Alexei G. Myasnikov, and Vladimir Shpilrain. Open problems in combinatorial group theory. In *Combinatorial and geometric group theory*, pages 1–38. American Mathematical Society, 2002.

4 Martin Beaudry, Pierre McKenzie, Pierre Péladeau, and Denis Thérien. Finite monoids: From word to circuit evaluation. *SIAM Journal on Computing*, 26(1):138–152, 1997.

5 David J. Buckley and Derek F. Holt. The conjugacy problem in hyperbolic groups for finite lists of group elements. *International Journal of Algebra and Computation*, 23(5):1127–1150, 2013.

6 Moses Charikar, Eric Lehman, April Lehman, Ding Liu, Rina Panigrahy, Manoj Prabhakaran, Amit Sahai, and Abhi Shelat. The smallest grammar problem. *IEEE Transactions on Information Theory*, 51(7):2554–2576, 2005.

7 François Dahmani and Vincent Guirardel. The isomorphism problem for all hyperbolic groups. *Geometric and Functional Analysis*, 21(2):223–300, 2011.

8 Max Dehn. Über unendliche diskontinuierliche Gruppen. *Mathematische Annalen*, 71:116–144, 1911.

9 Volker Diekert, Jürn Laun, and Alexander Ushakov. Efficient algorithms for highly compressed data: the word problem in Higman's group is in P. *International Journal of Algebra and Computation*, 22(8), 2012.

10 Will Dison, Eduard Einstein, and Timothy R. Riley. Ackermannian integer compression and the word problem for hydra groups. In *Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016*, volume 58 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

11 David B. A. Epstein, James W. Cannon, Derek F. Holt, Silvio V. F. Levy, Michael S. Paterson, and William P. Thurston. *Word Processing in Groups*. Jones and Bartlett, 1992.

12 David B. A Epstein and Derek F. Holt. The linearity of the conjugacy problem in word-hyperbolic groups. *International Journal of Algebra and Computation*, 16(2):287–306, 2006.

13 Elizaveta Frenkel, Andrey Nikolaev, and Alexander Ushakov. Knapsack problems in products of groups. *Journal of Symbolic Computation*, 74:96–108, 2016.

14 Moses Ganardi, Daniel König, Markus Lohrey, and Georg Zetzsche. Knapsack problems for wreath products. In *Proceedings of the 35th Symposium on Theoretical Aspects of Computer Science, STACS 2018*, volume 96 of *LIPIcs*, pages 32:1–32:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

15 Max Garzon and Yechezkel Zalcstein. The complexity of Grigorchuk groups with application to cryptography. *Theoretical Computer Science*, 88(1):83–98, 1991.

16 Mikhail Gromov. Hyperbolic groups. In S. M. Gersten, editor, *Essays in Group Theory*, number 8 in MSRI Publ., pages 75–263. Springer, 1987.

17 Christoph Haase. *On the complexity of model checking counter automata*. PhD thesis, University of Oxford, St Catherine's College, 2011.

18 Christian Hagenah. *Gleichungen mit regulären Randbedingungen über freien Gruppen*. PhD thesis, University of Stuttgart, 2000.

19 Frédéric Haglund and Daniel T. Wise. Coxeter groups are virtually special. *Advances in Mathematics*, 224(5):1890–1903, 2010.

20 Nico Haubold, Markus Lohrey, and Christian Mathissen. Compressed decision problems for graph products of groups and applications to (outer) automorphism groups. *International Journal of Algebra and Computation*, 22(8), 2013.

21 Derek F. Holt. Word-hyperbolic groups have real-time word problem. *International Journal of Algebra and Computation*, 10:221–228, 2000.

22 Derek F. Holt, Markus Lohrey, and Saul Schleimer. Compressed decision problems in hyperbolic groups. *CoRR*, abs/1808.06886, 2018. URL: `https://arxiv.org/abs/1808.06886`.

23 Derek F. Holt, Sarah Rees, and Claas E. Röver. *Groups, Languages and Automata*, volume 88 of *London Mathematical Society Student Texts*. Cambridge University Press, 2017.

24 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing, STOC 1997*, pages 220–229. ACM, 1997.

25 Martin Kassabov and Francesco Matucci. The simultaneous conjugacy problem in groups of piecewise linear functions. *Groups, Geometry, and Dynamics*, 6(2):279–315, 2012.

26 Daniel König and Markus Lohrey. Evaluation of circuits over nilpotent and polycyclic groups. *Algorithmica*, 80(5):1459–1492, 2018.

27 Daniel König, Markus Lohrey, and Georg Zetzsche. Knapsack and subset sum problems in nilpotent, polycyclic, and co-context-free groups. In *Algebra and Computer Science*, volume 677 of *Contemporary Mathematics*, pages 138–153. American Mathematical Society, 2016.

28 Markus Lohrey. Word problems and membership problems on compressed words. *SIAM Journal on Computing*, 35(5):1210 – 1240, 2006.

29 Markus Lohrey. Algorithmics on SLP-compressed strings: A survey. *Groups Complexity Cryptology*, 4(2):241–299, 2012.

30 Markus Lohrey. *The Compressed Word Problem for Groups*. SpringerBriefs in Mathematics. Springer, 2014.

31 Markus Lohrey. Knapsack in hyperbolic groups. In *Proceedings of the 12th International Conference on Reachability Problems, RP 2018*, volume 11123 of *Lecture Notes in Computer Science*, pages 87–102. Springer, 2018.

32 Markus Lohrey and Georg Zetzsche. Knapsack in graph groups. *Theory of Computing Systems*, 62(1):192–246, 2018.

33 Jeremy Macdonald. Compressed words and automorphisms in fully residually free groups. *International Journal of Algebra and Computation*, 20(3):343–355, 2010.

34 Jeremy MacDonald, Alexei G. Myasnikov, and Denis Ovchinnikov. Low-complexity computations for nilpotent subgroup problems. *CoRR*, abs/1706.01092, 2017. URL: `https://arxiv.org/abs/1706.01092`.

35 Alexei G. Myasnikov, Andrey Nikolaev, and Alexander Ushakov. Knapsack problems in groups. *Mathematics of Computation*, 84:987–1016, 2015.

36 Alexei G. Myasnikov, Alexander Ushakov, and Dong Wook Won. The word problem in the Baumslag group with a non-elementary Dehn function is polynomial time decidable. *Journal of Algebra*, 345(1):324–342, 2011.

37 Alexei G. Myasnikov, Alexander Ushakov, and Dong Wook Won. Power circuits, exponential algebra, and time complexity. *International Journal of Algebra and Computation*, 22(6), 2012.

38 Alexander Yu. Ol'shanskii. Almost every group is hyperbolic. *International Journal of Algebra and Computation*, 2(1):1–17, 1992.

39 Wojciech Plandowski. Testing equivalence of morphisms on context-free languages. In *Proceedings of the 2nd Annual European Symposium on Algorithms, ESA 1994*, volume 855 of *Lecture Notes in Computer Science*, pages 460–470. Springer, 1994.

40 Saul Schleimer. Polynomial-time word problems. *Commentarii Mathematici Helvetici*, 83(4):741–765, 2008.

41 Stephan Waack. The parallel complexity of some constructions in combinatorial group theory. *Journal of Information Processing and Cybernetics EIK*, 26:265–281, 1990.

42 Daniel T. Wise. Research announcement: the structure of groups with a quasiconvex hierarchy. *Electronic Research Announcements in Mathematical Sciences*, 16:44–55, 2009.